Drupal In A Day

Sunil Movva (UAH)
Jerry Pan (ORNL DAAC)
Giri Palanisamy (ORNL DAAC)

Agenda

Part 1

1. What is Drupal and why you care
2. Software Installation
3. Building a Drupal website
4. Administering a Drupal site
5. Drupal Theme Concepts

Part 2

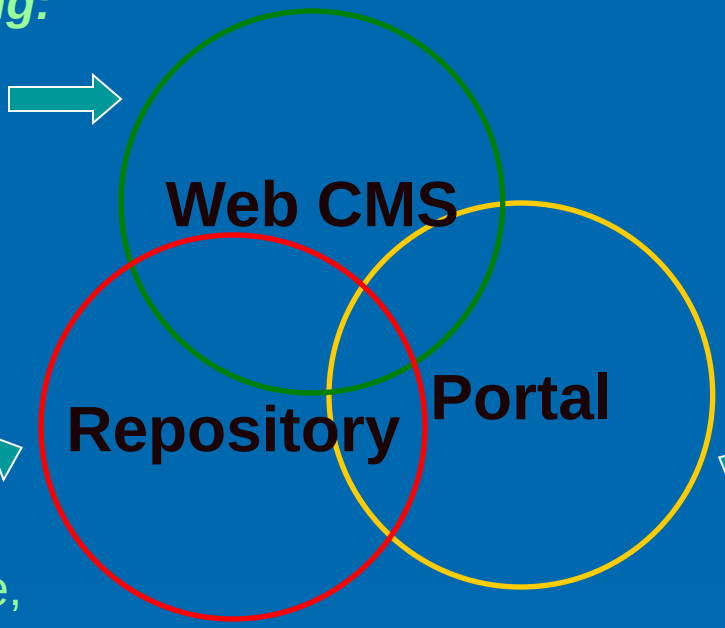
6. Contributed modules
7. Creating custom content types
8. An introduction to module development

1. What is Drupal

   (and why you care)

Content Management Frameworks

Web Publishing:
Drupal, Plone,
WordPress, …



Preservation:
Fedora, DSpace,
Alfresco, …

Integration:
Liferay,
JBoss Portal, …
(Java standards)

Drupal

- A popular Web Content Management System (CMS), open source (GPL), ~10 years old (Dries Buytaert)
- Modular and extensible design
- Large user and developer community, thousands of contributed modules
- Currently at version 6.17, version 7 upcoming
- Framework written in PHP, runs on major OS platforms

Core Features

- Content management
  - With admin user interface
  - Separation of content from view*
  - Custom content types
  - Versioning
  - Taxonomy support
  - Search support
- Template & theme system
  - Many faces, same content
- User management
  - User authentication and role-based authorization

* Similar to Model View Controller (MVC) design pattern, although there is suggestion that Drupal is closer to Presentation Abstraction Control (PAC) pattern

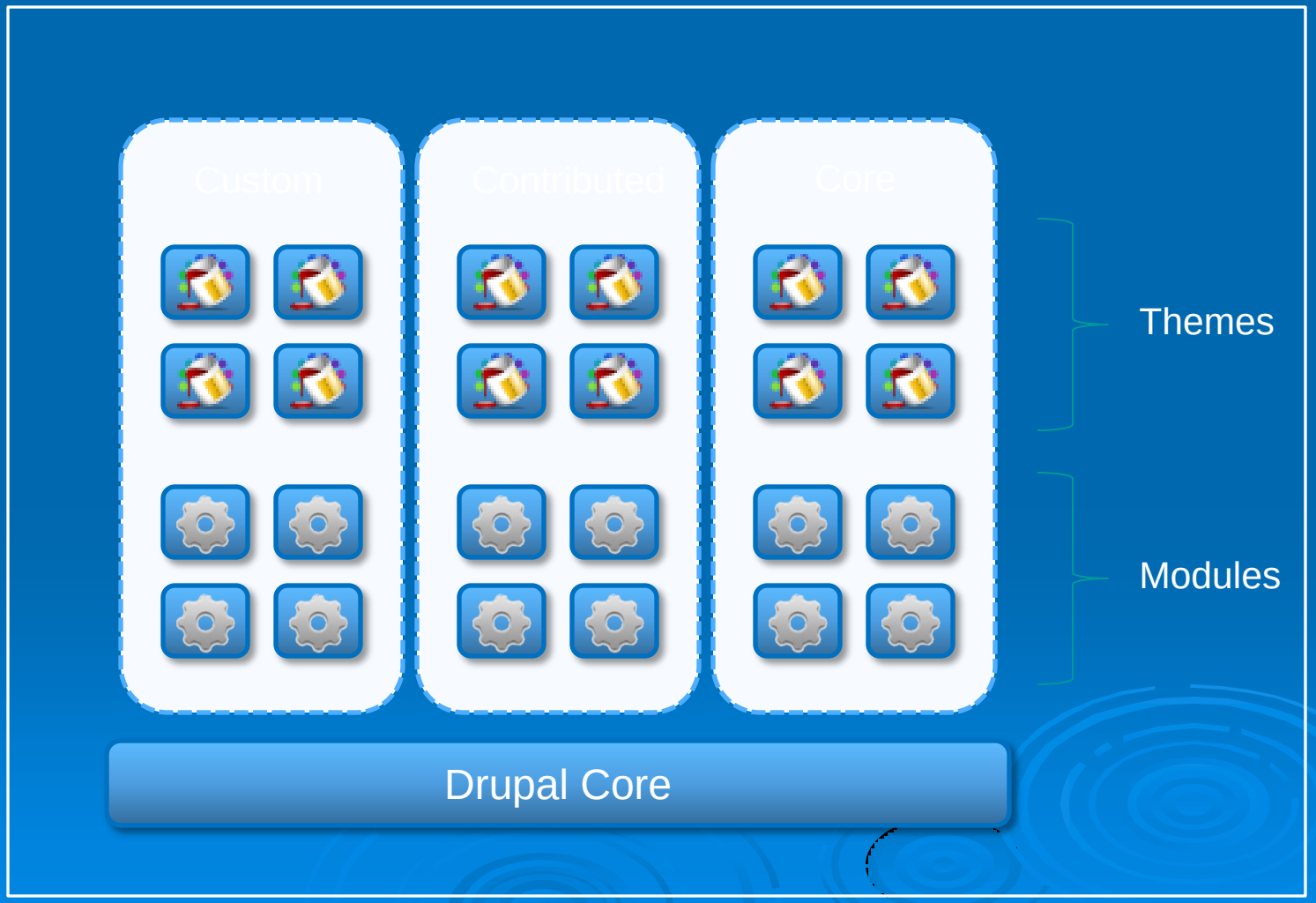
Core Concepts (D.6)

- Administration Interface: build and administer a site – enable/disable/configure/change various components
- Content type: content is typed (string marker)
- Node: underline storage units for all types (“base class”)
- Theme: assembles HTML, scripts to display content
- Block: information placed on a display location
- Users and Roles: users stored in table, password hashed, not a node; a role = a group of users with the same privileges

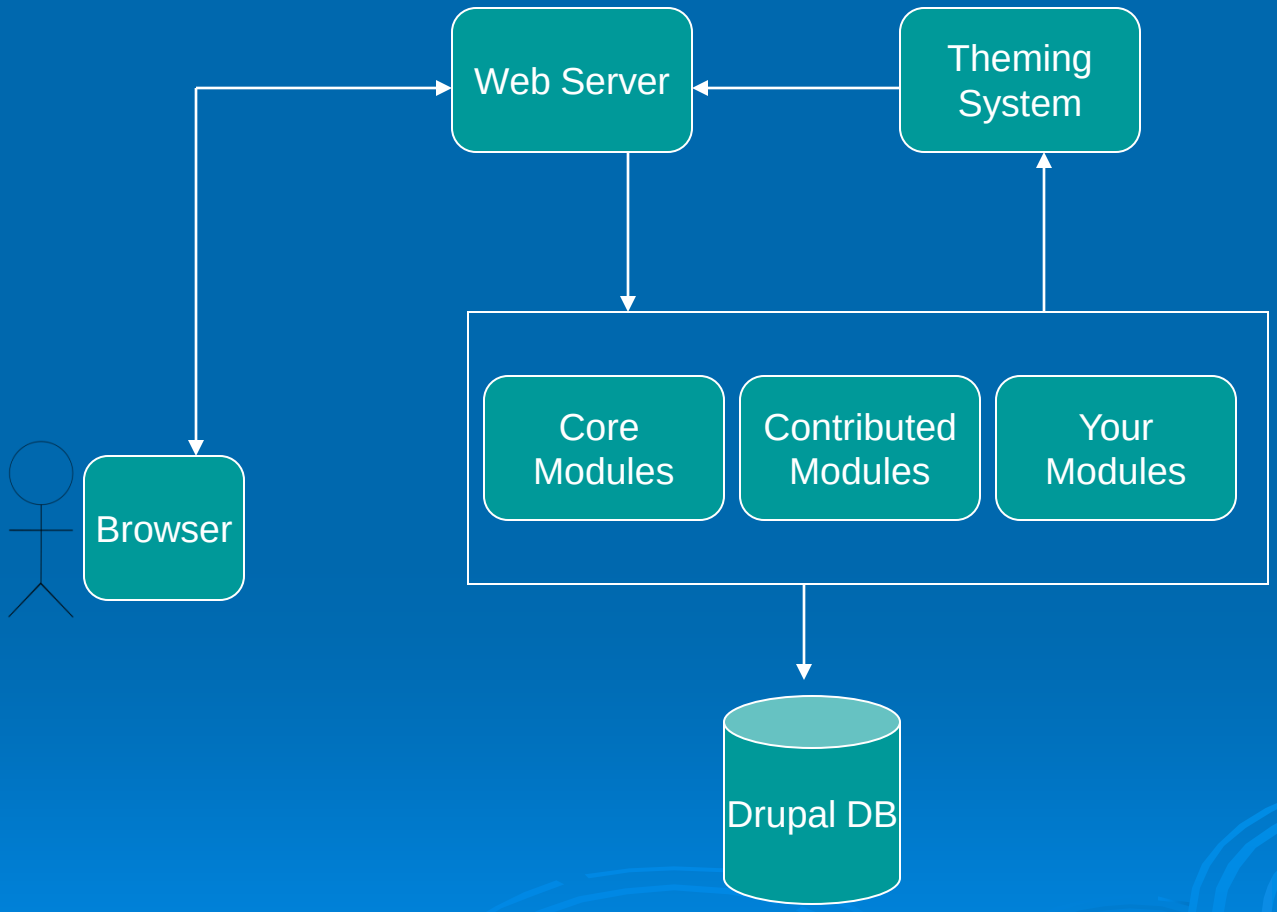
Core Concepts (cont.)

- Files: attachments to nodes (not stored in tables)
- Comments: comments on a node, not nodes themselves
- Modules: building blocks of Drupal, provides functionality
- Hooks: Events API, or callback. A module implements certain hooks to provide functionalities
- Navigation system: a navigation block with menus, primary links, secondary links
- Views: a smart query builder that defines how content are extracted and presented

Drupal System



How It Works – Abstract View



Uses of Drupal

- Drupal can be used for most web site applications, including
  - Data/Metadata management
  - Corporate web sites
  - Community websites (content portals)
  - Intranet applications
  - Personal web sites or blogs
  - Polls
  - Collaborative websites
  - Forums
  - News site
  - Podcasting
  - Picture galleries
  - Document management
  - Aficionado sites
  - E-commerce applications
  - Resource directories
  - Social Networking sites
- Better suitable for more complex web sites, or sites flexible to evolve, see:
  - http://drupal.org/node/346217
- Multi-sites supported

Drupal Advantages

- Powerful: more than likely modules exists for your needs
- Extensible: develop your own modules with Drupal APIs (hooks)
- Flexible: lots of options for customizing to your needs (choose modules, and customizable look and feel)
- Large community: the network effect, growing ecosystem: training/support options, available talents
- Security: sound design, alert options, and update system

What to watch out

- Learning curve and training cost
  - Unique Drupal concepts, overwhelming features
  - Third-party training & service costs (true for any software!)
- Migration existing system into Drupal
  - Can be Complex
  - Unique challenges case by case
- Lack of control, particularly on the core system
- Major version upgrade (e.g. from 6.x to 7.x) may be costly
- Watch out for too many modules: be careful introducing a new module, document its purpose

Resources

- http://www.drupal.org/
  - Download core, contributed module & themes
  - Documentations, tutorials, tips
  - List of training, consulting services
  - much more
- http://www.meetup.com/ for local Drupal Groups
- Conferences & Seminars
  - DrupalCon (2/year, e.g., http://sf2010.drupal.org/)
  - Do It with Drupal (~1/year)

Resources (cont.)

- Books:
  - “Pro Drupal Development” second edition, by John K. VanDyk (Apress)
  - “Building powerful and robust websites with Drupal 6”, by David Mercer (PACKT Publishing)
  - “Using Drupal”, by Angela Byron and others (O’Reilly)
  - “Drupal 6 Themes”, by Ric Shreves (PACKT Publishing)

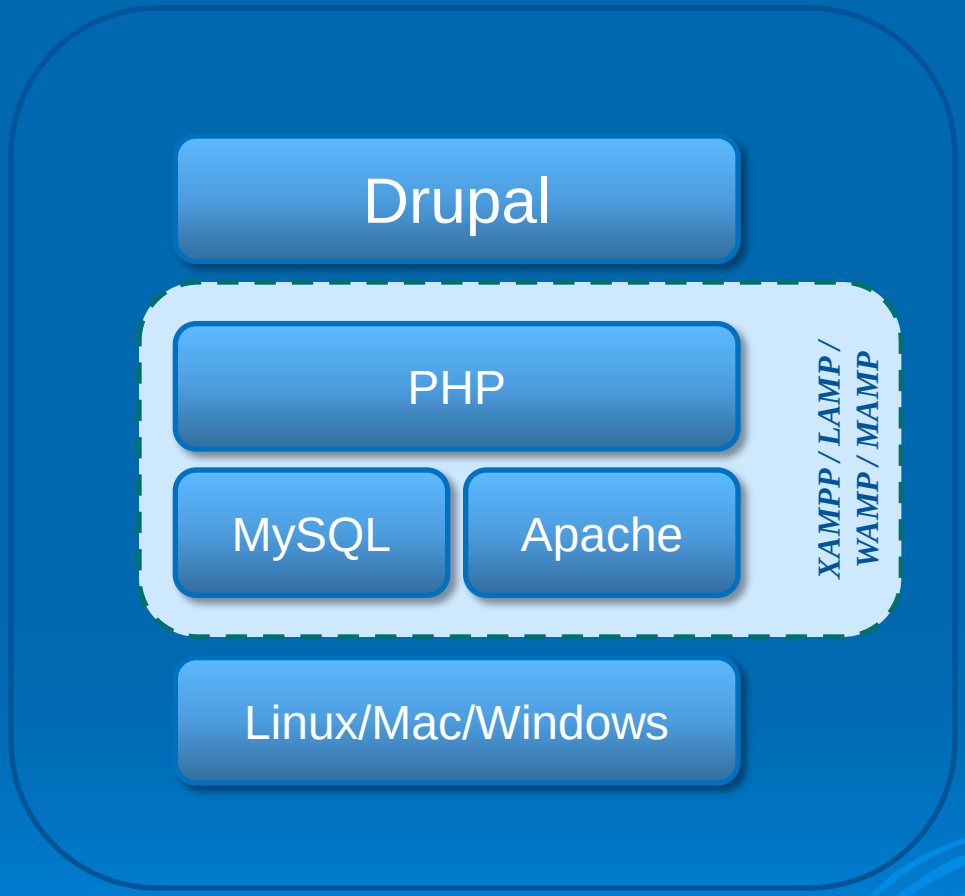
Resources (cont.)

- http://drupalmodules.com/
  - Drupal module review and download
  - Easy search and browse
- http://themegarden.org/drupal6/
  - Browse Drupal themes as actually used
- http://mustardseedmedia.com/podcast
  http://drupal.org/node/124318
  - Videos, podcast on Drupal topics

Resources (cont.)

- http://www.easypagesaver.com/feed-item/76595
  - Drupal news
- http://www.lullabot.com/ and http://learnbythedrop.com/
  - Training services
- http://www.acquia.com/
  - Support, consulting services

2. Software Installation



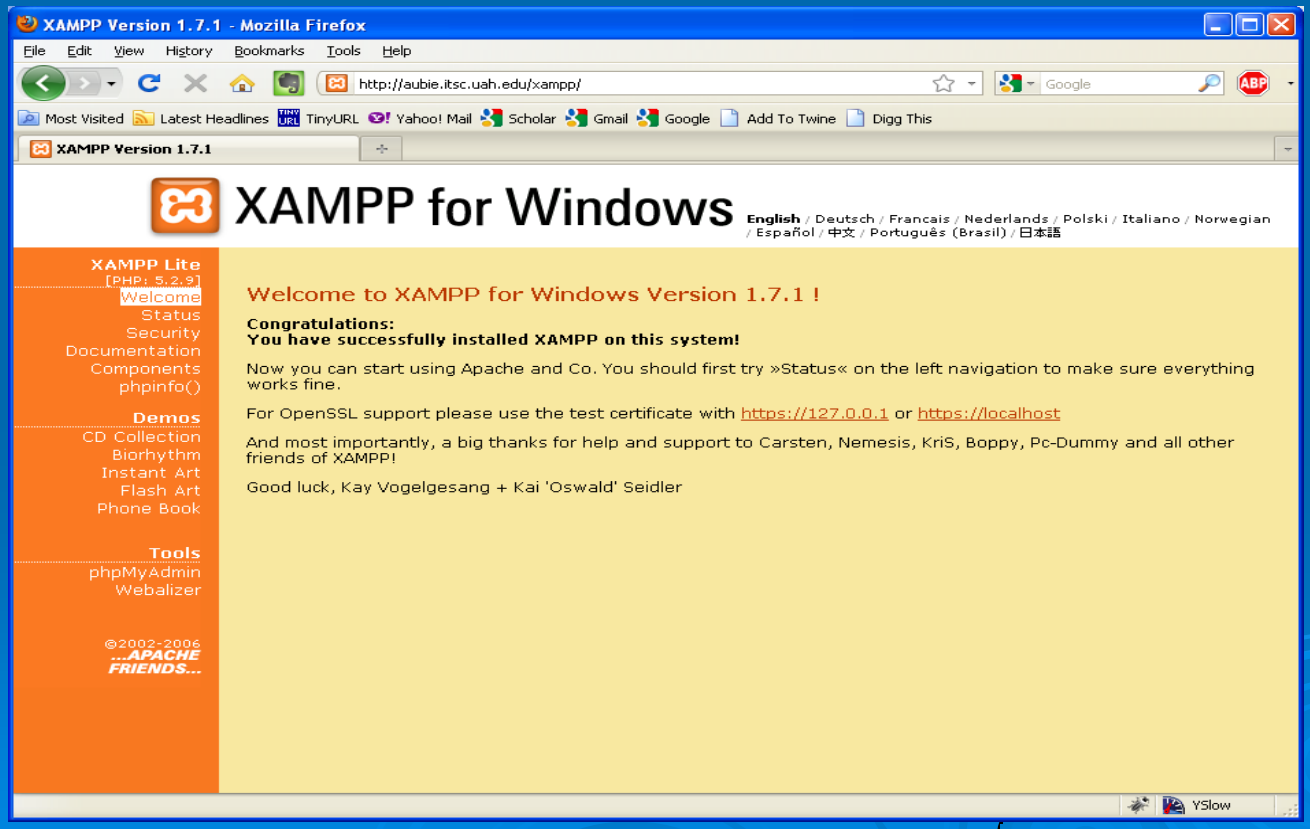
Drupal Stack

Software Installation (cont.)

- Download XAMPP : http://www.apachefriends.org/en/xampp.html
- Download Drupal : http://drupal.org/project/drupal
- Install XAMPP
- Start XAMPP control panel
- Start MySql and Apache services from the control panel

XAMPP installation

- Navigate to http://localhost/xampp/



Database setup

- Go to MyPhpAdmin
- Create a new database ‘mysitedb’
- Create a new user ‘mysiteweb’ (copy the password)
- Grant all privileges

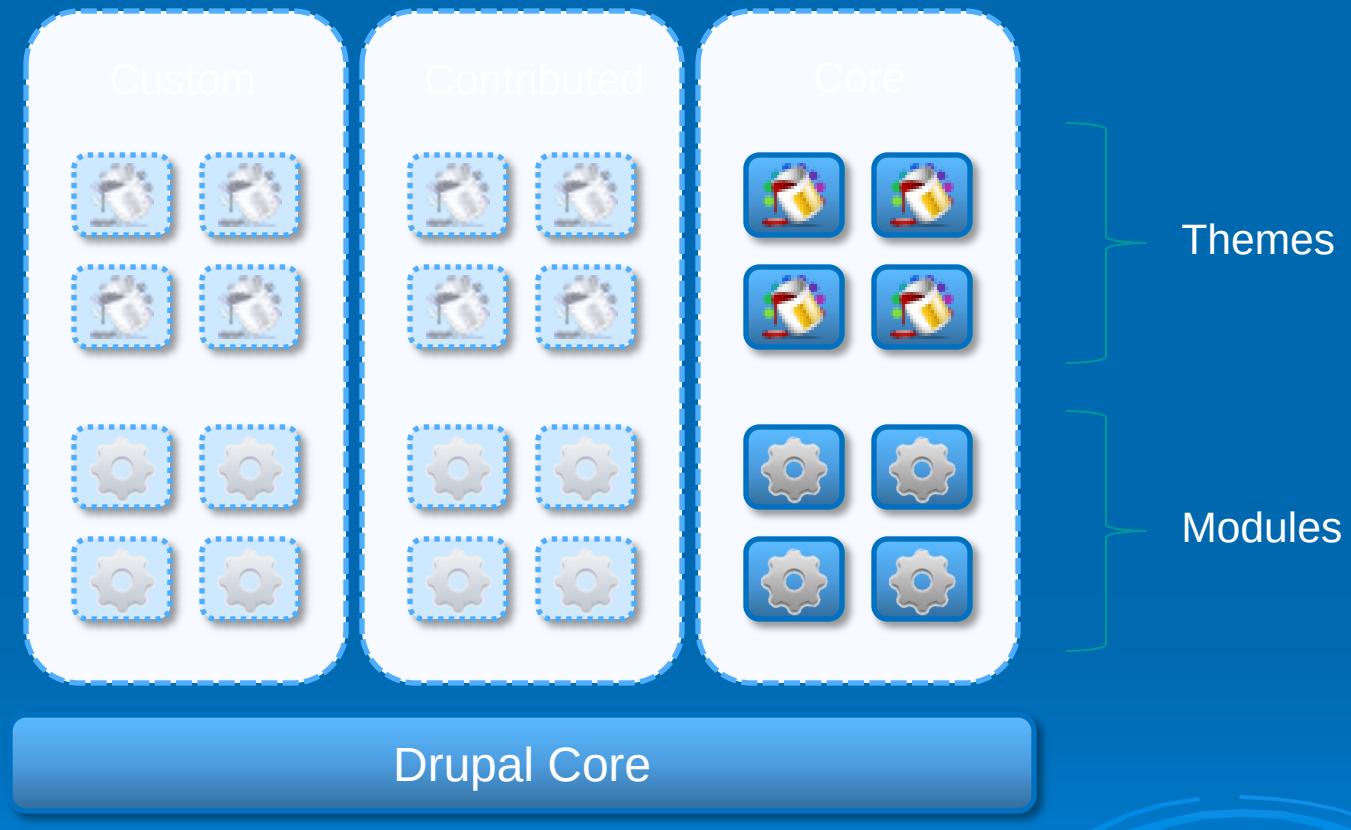
Drupal Installation

- Extract drupal-6.xx.tar.gz to <xampp>/htdocs/
- Rename drupal-6.xx to mysite
- Copy default.settings.php to settings.php in the mysite/sites/default/ folder
- Give all privileges to settings.php file

Drupal Installation (Cont.)

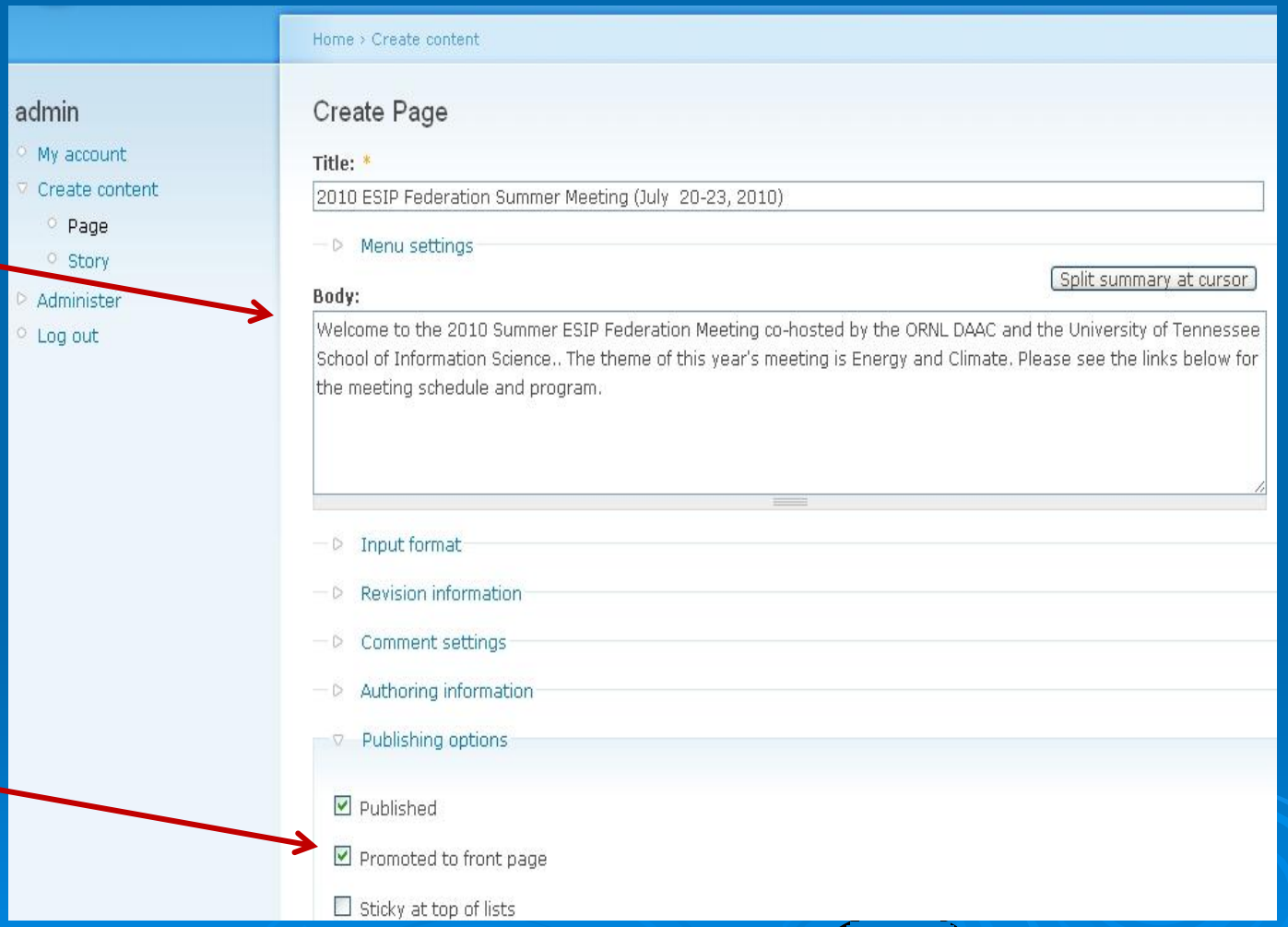
- Navigate to http://localhost/mysite/ in your web browser and follow the instructions
- Check the files directory and make sure it has write permissions

Drupal Installation (Cont.)



3. Building Content

Create Content



4. Site Administration

Components of a Drupal Site

5 Header

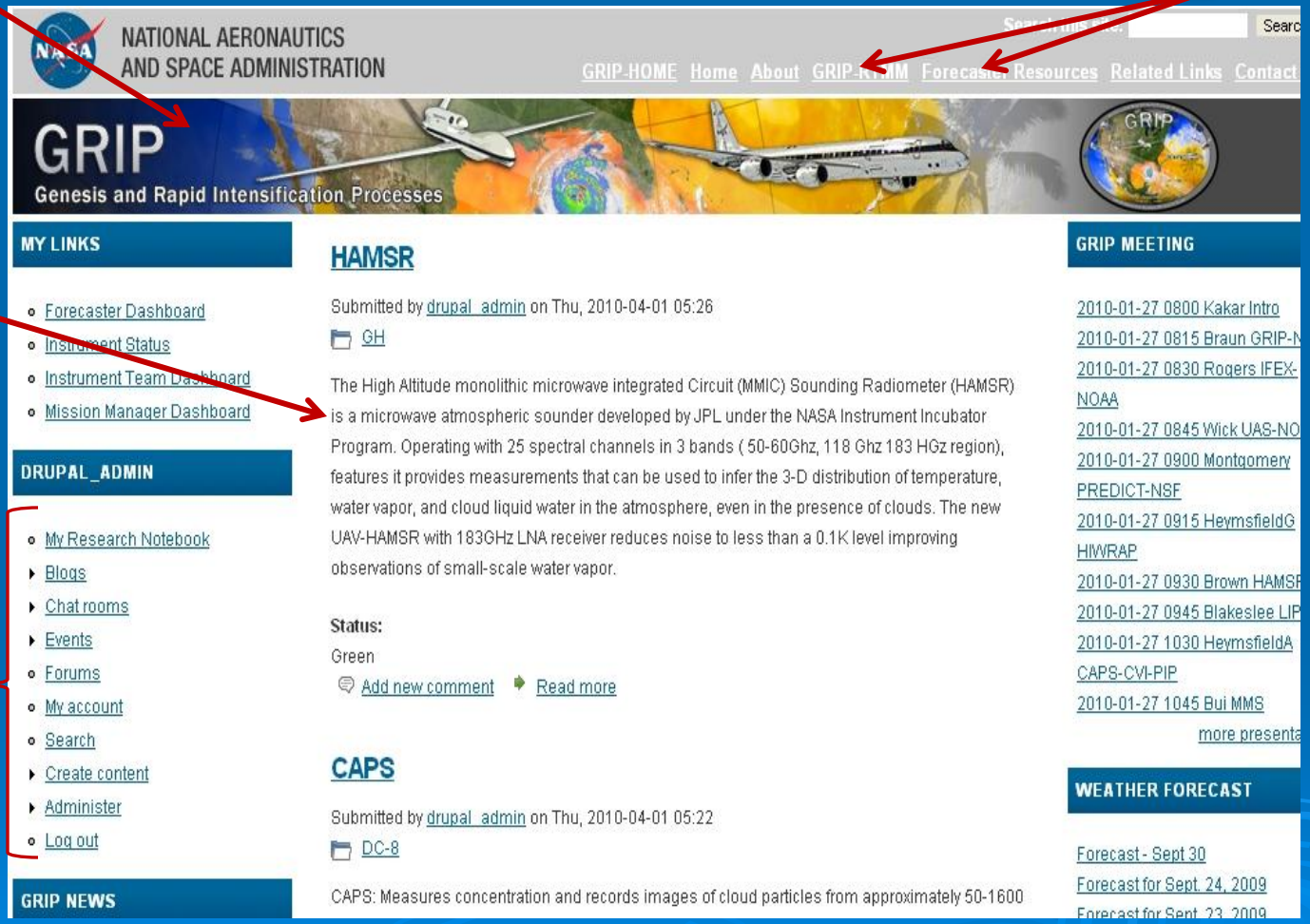
2 Primary Menus

4 Content Area

3 Blocks

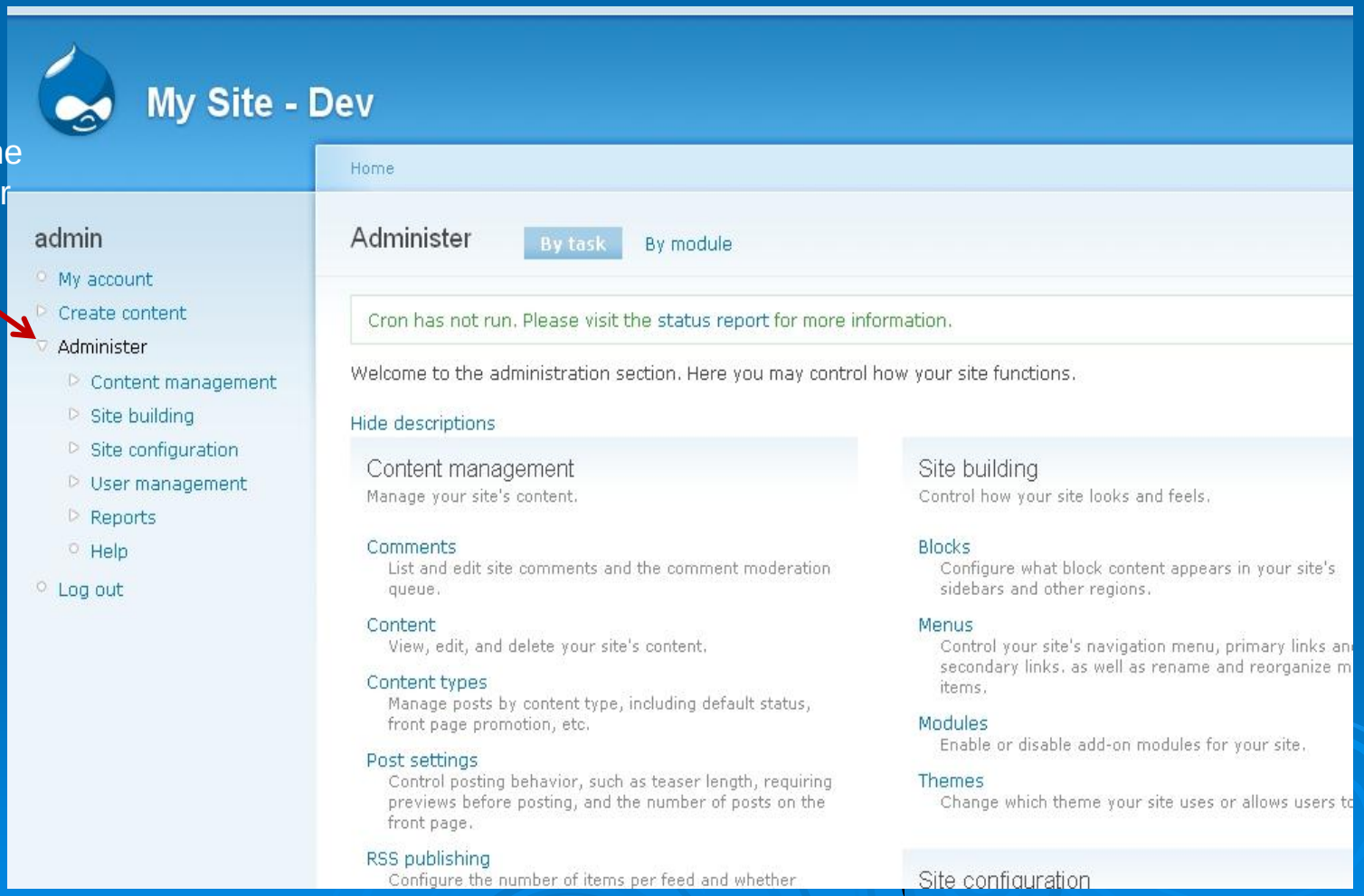
1 Navigation Menu

6 Footer



Site Administration

Click on the Adminster Menu



Site Administration (Cont.)

- Site information
- Enable/Disable Modules
- Configure Blocks
- Menus

Backup/Restore

- Backup often: database, file system with Cron job, Task Scheduler, etc.
- Many tools possible, examples
  - Database: mysqldump, mysql, PhpMyAdmin
  - File system: tar, zip
- Backup: backup database, backup files/directories
- Restore: turn site offline, re-populate database from backup, restore file system from backup

Update for bugs, security

- Setup new version alert:
  - *Administer* > *Reports* > *Available updates* > *Settings*
- Drupal Core upgrade: minor version
- Module upgrade

- Demo: update Drupal core (or a contributed module)
  - Turn site off-line (if it has existing users)
  - Note: if browser session ends, log back in with ?q=user/login
  - Backup old site files to a separate location
  - Download/Install new core, or module
  - Restore files from backup: /sites/* (for core update only)
  - Run update.php (always run it for any update)
  - Turn site back online

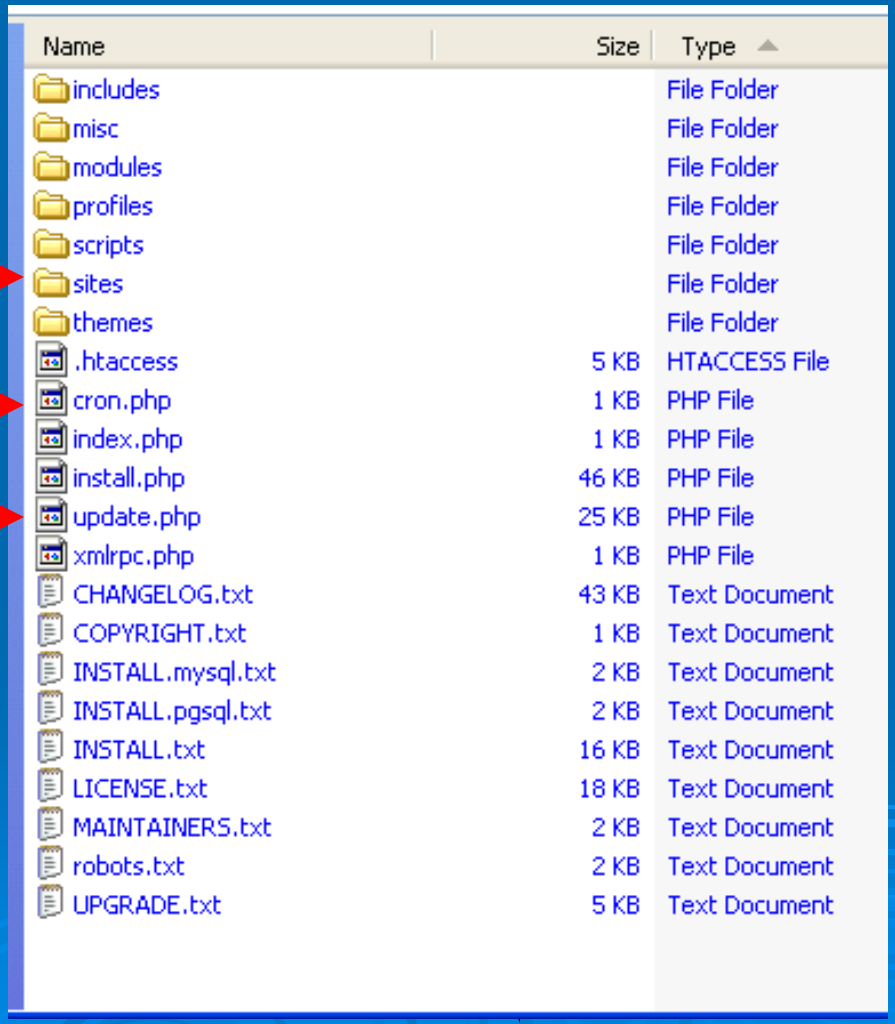
https://esdora.ornl.gov/drupal-test/admin

Drupal Files and Directories

Sites: customizations
"sites/all/modules"
"sites/all/themes"

Maint. cron job

Update scripts



Migration To Different Host

- Reasons: dev site, hardware upgrade
- Update first, if needed
- Document any differences in software stack (minimal difference desirable)
- Turn off clean URLs
- Consider to turn site off-line (show no errors)
- Backup
- Setup database and database user on new host
- Restore

Users and Permissions

- Allows:
  - Create Users (Super user)
  - Create various Roles and Assign users to different roles
  - Setup Permissions
  - Define: Access Rules, User Settings.

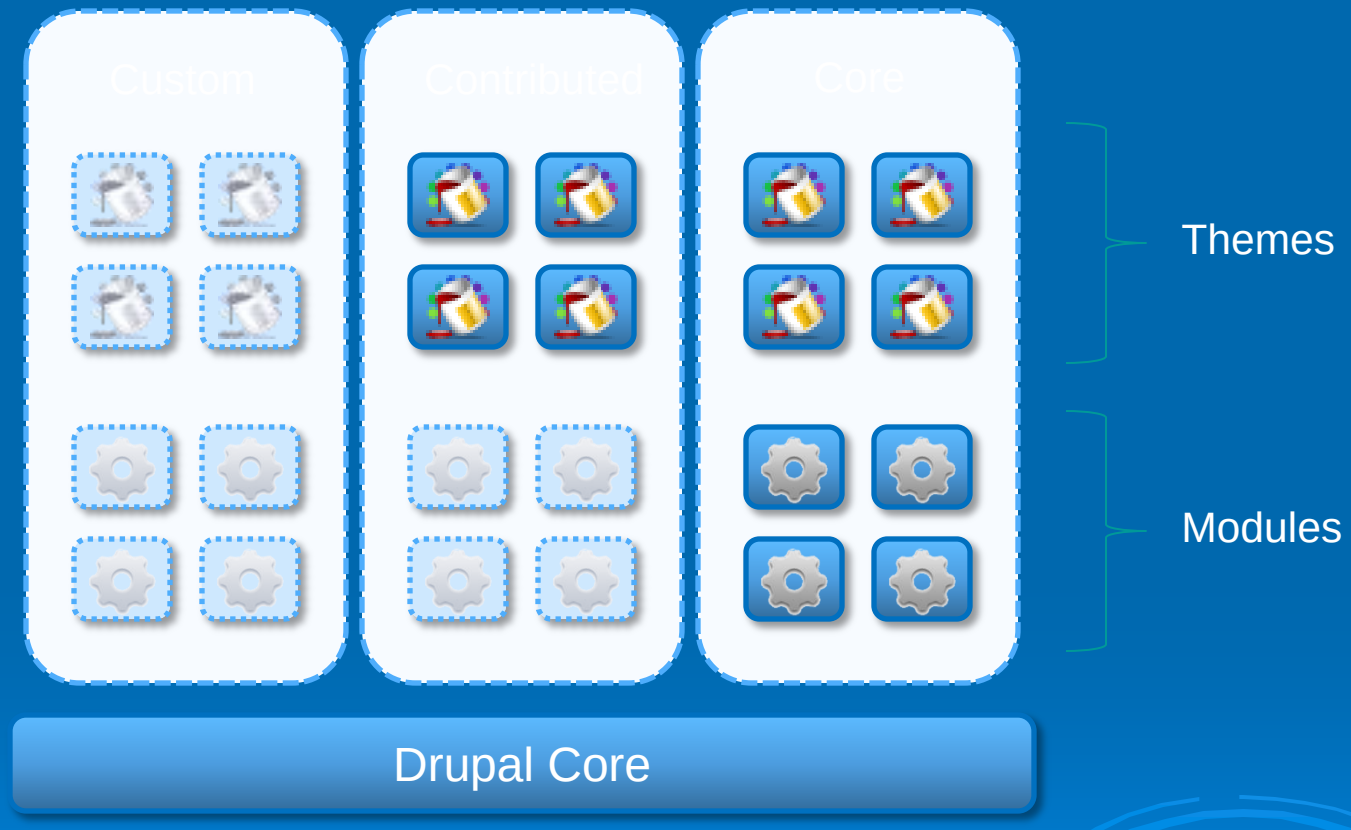
  Demo….

5. Introduction to Drupal Themes

Introduction to Drupal Theme

- Themes allow you to change the look and feel of your Drupal site
  - Default themes (demo)
  - Download more themes from the Web
  - Contributed themes: http://drupal.org/project/Themes
  - Preview some themes: http://themegarden.org/drupal6/
  - Customize existing themes or your own theme development

Drupal Themes(Cont.)



Drupal in a day, Part 2

Agenda:

6. Contributed modules
7. Creating custom content types
8. An introduction to module development

6. Contributed Modules

Contributed Modules

- What is it?
- How to find & select
  - http://drupal.org/project/Modules
- Usage statistics
  - http://drupal.org/project/usage
- Installation
  - Make sure to download the correct version
  - unzip and extract the code to /sites/all/modules/
- Contribute back to the Drupal community
  - http://drupal.org/node/23789

Some Useful Modules

1) Site Management:

- Administration menu, Anti -SPAM modules (Captcha, Mollum), Devel, Panels, Pathauto, XML Sitemap, Site Verification, Table Wizard, STORM, Event

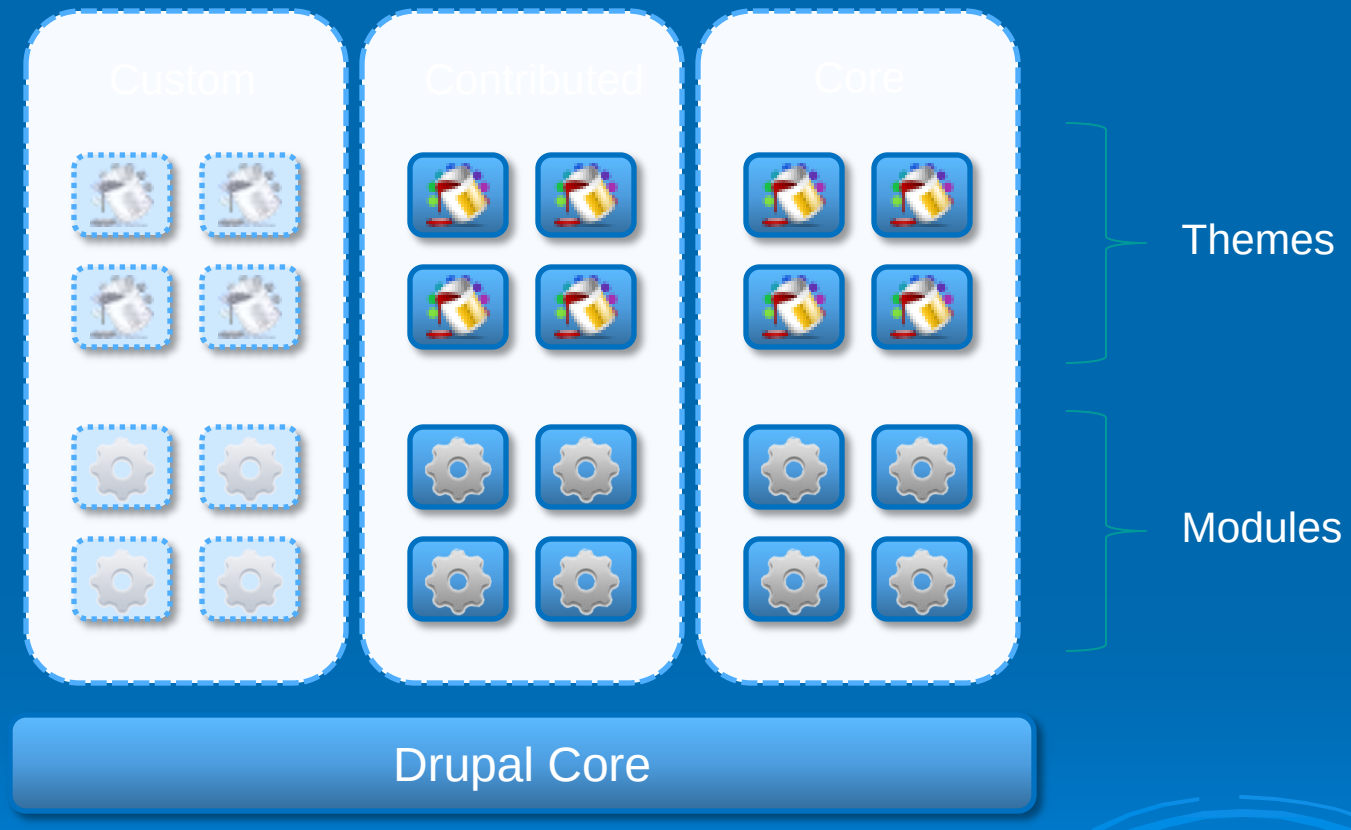
2) Content Creation:

- Content Taxonomy, Date, Token, ImageCache, ImageField, Jquery

Modules Useful for Science-Oriented Site

- **Content Types:** FAQ, Biblio, Data, Location, Forum, Blog, Poll, XML Content, RDF, LDAP
- **Mapping/OGC:** GMap, Nice Map, Open Layers, KML, Google Visualization API
- **Services:** OAI-PMH, Google Analytics
- **Other Science Oriented:**
  - Islandora Module (Fedora Commons Repository, digital object repository)
  - Metadata Editor work (using many modules)

Contributed Modules(Cont.)



7. Creating custom content types: CCK and Views

Content Construction Kit, Views

- Custom Content Type: created by you
- Views Module: a smart query builder that defines how content are extracted and presented
- CCK and View modules
  - Must have for moderately complex site
  - CCK becoming core in Drupal 7

Install, Use CCK-View

- Find modules to download (*do it with us*)
  - http://drupal.org/project/modules
  - Filter by 6.x
  - Search for Content Construction Kit (CCK), FileField, Views …
  - or, http://drupal.org/project/[module_name]
- Install Modules

CCK, View (cont.)

- Content type Admin UI
  - “News” : without CCK
  - Enable CCK, show default fields
  - Enable FileField
  - “My Document” type with CCK
- Views Admin UI
  - Creates complex SQL queries to extract content for display, from the UI
  - May take some time to get use to
  - “Advanced Help Module” for help

8. An introduction to module development

Hooks

- Allow modules to interact with the Drupal core.
- Hooks provide an opportunity for your module to act upon a drupal event.
- Hooks are php functions.
- Eg: foo_bar() where foo is the name of your module and bar is the name of the hook.
- http://api.drupal.org/

Anatomy of a Drupal Module

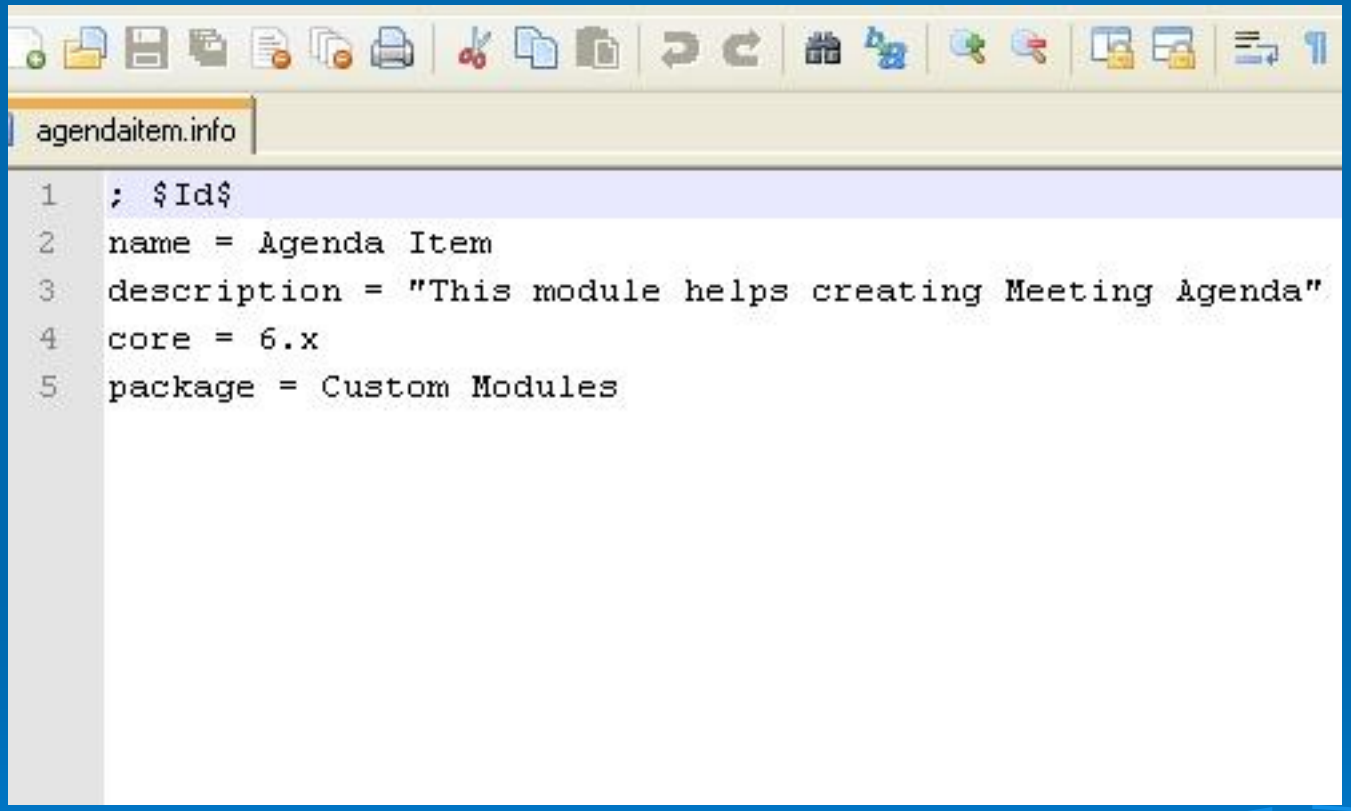
Drupal Modules have three files

- INFO file
- INSTALL file – optional
- MODULE file

INFO File

- Tells drupal about your module
- Syntax is similar to .ini files
  - name = value pairs
- Details:
  - name : Human readable name of your module
  - description : Short description of what the module does
  - core: Drupal core compatibility
  - dependencies (Optional) : What other modules need to be enabled
  - package (Optional) : Grouping modules in the admin page
- http://drupal.org/node/206756

agendaitem.info



Is it a node?

- Does your module need to define node types?
- Advantages of using node system
  - Integrates well with the core
  - Integrates well with contributed modules thus getting lot of functionality

Agenda Item Content-type

- Needs two additional fields apart from the title and the body from the node type.
  - Presenter
  - Timeslot
- Store the additional data in a new database table

INSTALL File

- Tells drupal to do certain things when this module is installed or uninstalled
  - Typically, creation/deletion of additional database tables

agendaitem.install

agendaitem.install

```
4  /*
5  *  @file agendaitem.install
6  *  Install file for agendaitem module
7  */
8
9  /*
10 *Implements hook_install
11 */
12 function agendaitem_install() {
13     drupal_install_schema('agendaitem');
14 }
15
16 /*
17 *Implements hook_uninstall
18 */
19 function agendaitem_uninstall() {
20     drupal_uninstall_schema('agendaitem');
21 }
22
23 /**
24 * Implements hook_schema
25 */
26 function agendaitem_schema() {
27     $schema['agendaitem'] = array (
28         'fields' => array(
29             'vid' => array('type' => 'int', 'unsigned' => TRUE, 'not null' => TRUE, 'default' => 0),
30             'nid' => array('type' => 'int', 'unsigned' => TRUE, 'not null' => TRUE, 'default' => 0),
31             'presenter' => array('type' => 'varchar', 'length' => 255, 'not null' => TRUE, 'default' => ''),
32             'timeslot' => array('type' => 'varchar', 'length' => 255, 'not null' => TRUE, 'default' => ''),
33         ),
34         'primary key' => array('vid', 'nid')
35     );
36     return $schema;
37 }
38
39
40
```

Module File (Defining new content-types)

```
1  /*
2  *    Implementation of hook_node_info()
3  *
4  */
5
6  function agendaitem_node_info(){
7      return array(
8          'agendaitem' => array(
9              'name' => t('Agenda Item'),
10             'module' => 'agendaitem',
11             'description' => t("This is an agenda node type."),
12             'has_title' => TRUE,
13             'title_label' => t('Session Title'),
14             'has_body' => TRUE,
15             'body_label' => t('Abstract'),
16         ),
17     );
18 }
19
```

agendaitem.module (Node operations)

```php
/**
 * Implementation of hook_access().
 *
 * Node modules may implement node_access() to determine the operations
 * users may perform on nodes. This example uses a very common access pattern.
 */
function agendaitem_access($op, $node, $account) {
  if ($op == 'create') {
    return user_access('create agendaitem content', $account);
  }

  if ($op == 'update') {
    if (user_access('edit any agendaitem content', $account) ||
        (user_access('edit own agendaitem content', $account) && ($account->uid == $node->uid))) {

      return TRUE;
    }
  }

  if ($op == 'delete') {
    if (user_access('delete any agendaitem content', $account) ||
        (user_access('delete own agendaitem content', $account) && ($account->uid == $node->uid))) {

      return TRUE;
    }
  }
}
```

agendaitem.module (Showing Permissions on adminpage)

```
1  /**
2   * Implementation of hook_perm().
3   *
4   * Since we are limiting the ability to create new nodes to certain users,
5   * we need to define what those permissions are here. We also define a permission
6   * to allow users to edit the nodes they created.
7   */
8  function agendaitem_perm() {
9    return array(
10     'create agendaitem content',
11     'delete own agendaitem content',
12     'delete any agendaitem content',
13     'edit own agendaitem content',
14     'edit any agendaitem content',
15   );
16 }
17
```

agendaitem.module (Collecting data from users)

```php
/**
 * Implementation of hook_form().
 *
 * Now it's time to describe the form for collecting the information
 * specific to this node type. This hook requires us to return an array with
 * a sub array containing information for each element in the form.
 */
function agendaitem_form(&$node) {
  // The site admin can decide if this node type has a title and body, and how
  // the fields should be labeled. We need to load these settings so we can
  // build the node form correctly.
  $type = node_get_types('type', $node);

  if ($type->has_title) {
    $form['title'] = array(
      '#type' => 'textfield',
      '#title' => check_plain($type->title_label),
      '#required' => TRUE,
      '#default_value' => $node->title,
      '#weight' => -5
    );
  }

  if ($type->has_body) {
    $form['body_field'] = node_body_field($node, $type->body_label, $type->min_word_count);
  }

  // Now we define the form elements specific to our node type.
  $form['presenter'] = array(
    '#type' => 'textfield',
    '#title' => t('Presenter'),
    '#default_value' => isset($node->presenter) ? $node->presenter : '',
  );
  $form['timeslot'] = array(
    '#type' => 'textfield',
    '#title' => t('Timeslot'),
    '#default_value' => isset($node->timeslot) ? $node->timeslot : '',
  );

  return $form;
}
```

http://api.drupal.org/api/drupal/developer--topics--forms_api_reference.html/6

agendaitem.module (Store/update data in the database)

```
/**
 * Implementation of hook_insert().
 *
 * As a new node is being inserted into the database, we need to do our own
 * database inserts.
 */
function agendaitem_insert($node) {
  db_query("INSERT INTO {agendaitem} (vid, nid, presenter, timeslot) VALUES (%d, %d, '%s', '%s')",
           $node->vid, $node->nid, $node->presenter, $node->timeslot);
}

/**
 * Implementation of hook_update().
 *
 * As an existing node is being updated in the database, we need to do our own
 * database updates.
 */
function agendaitem_update($node) {
  // if this is a new node or we're adding a new revision,
  if ($node->revision) {
    agendaitem_insert($node);
  }
  else {
    db_query("UPDATE {agendaitem} SET presenter = '%s', timeslot = '%s' WHERE vid = %d",
             $node->presenter, $node->timeslot, $node->vid);
  }
}

/**
 * Implementation of hook_delete().
 *
 * When a node is deleted, we need to remove all related records from our table.
 */
function agendaitem_delete($node) {
  // Notice that we're matching all revision, by using the node's nid.
  db_query('DELETE FROM {agendaitem} WHERE nid = %d', $node->nid);
}

/**
 * Implementation of hook_nodeapi().
 *
```

agendaitem.module
(Loding the data back into node object)

```
167
168 /**
169  * Implementation of hook_load().
170  *
171  * Now that we've defined how to manage the node data in the database, we
172  * need to tell Drupal how to get the node back out. This hook is called
173  * every time a node is loaded, and allows us to do some loading of our own.
174  */
175 function agendaitem_load($node) {
176   $additions = db_fetch_object(db_query('SELECT presenter, timeslot FROM {agendaitem} WHERE vid = %d', $node->vid));
177   return $additions;
178 }
179
180 /**
```

agendaitem.module (Presenting to the user)

```
181  * Implementation of hook_view().
182  *
183  * This is a typical implementation that simply runs the node text through
184  * the output filters.
185  */
186 function agendaitem_view($node, $teaser = FALSE, $page = FALSE) {
187   $node = node_prepare($node, $teaser);
188   $node->content['myfields'] = array(
189     '#value' => theme('agendaitem_info', $node),
190     '#weight' => 1,
191   );
192   return $node;
193 }
194
195 /**
196  * Implementation of hook_theme().
197  *
198  * This lets us tell Drupal about our theme functions and their arguments.
199  */
200 function agendaitem_theme() {
201   return array(
202     'agendaitem_info' => array(
203       'arguments' => array('node'),
204     ),
205   );
206 }
207
208 /**
209  * A custom theme function.
210  *
211  * By using this function to format our node-specific information, themes
212  * can override this presentation if they wish. We also wrap the default
213  * presentation in a CSS class that is prefixed by the module name. This
214  * way, style sheets can modify the output without requiring theme code.
215  */
216 function theme_agendaitem_info($node) {
217   $output = '<div class="agendaitem-presenter">';
218   $output .= t('Presenter: %presenter', array('%presenter' => check_plain($node->presenter)));
219   $output .= '</div>';
220   $output .= '<div class="agendaitem-timeslot">';
221   $output .= t('Time Slot: %timeslot', array('%timeslot' => check_plain($node->timeslot)));
222   $output .= '</div>';
223
224   return $output;
225 }
```

Module Development Resources

- http://api.drupal.org/
- Best Practices: http://drupal.org/node/360052
- Useful Modules:
  - Trace : http://drupal.org/project/trace
  - Devel : http://drupal.org/project/devel
  - Coder: http://drupal.org/project/coder
- Useful API and other functions:
  - watchdog : http://api.drupal.org/api/function/watchdog/6
  - dprint_r : Devel module

Module Development Resources

- Development Enviroment:
  - Eclipse PDT with Zend debugger: http://drupal.org/node/723470
- Profiles:
  - Aquia: http://acquia.com/downloads
  - Open Atrium: http://openatrium.com/